

Часть III

Структурный подход в распознавании образов (I)

Разделы

Введение в синтаксический подход к распознаванию образов

Обзор подходов к задаче распознавания образов

Методы предобработки объектов

Языки описания образов

Автоматные языки и конечные автоматы

- └ Введение в синтаксический подход к распознаванию образов

- └ Обзор подходов к задаче распознавания образов

Разделы

Введение в синтаксический подход к распознаванию образов

- Обзор подходов к задаче распознавания образов

- Методы предобработки объектов

- Языки описания образов

Автоматные языки и конечные автоматы

Распознавание образов: дискриминантный подход

① Дискриминантный (признаковый) подход

Объекты характеризуются набором признаков и распознавание проводят разбиением пространства признаков на области.

Методы:

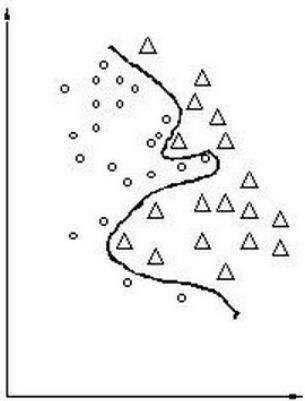
- ▶ метрические (NN, ...);
- ▶ разделяющие поверхности (SVM, ...);
- ▶ потенциальные функции;
- ▶ логические;
- ▶ информационные;
- ▶ коллективные решающие правила (*области компетенции, голосование, алгебраический подход*);
- ▶ ...

Математический аппарат — метрические теории, логические функции, теория информации, алгебра алгоритмов, ...

└ Введение в синтаксический подход к распознаванию образов

└ Обзор подходов к задаче распознавания образов

Распознавание образов: дискриминантный подход...



«Ленивые» и «не-ленивые»
алгоритмы распознавания

При этом игнорируется информация:
(1) структурная; (2) объект-признак

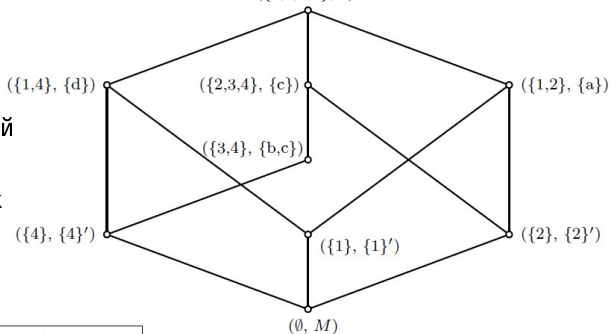
└ Введение в синтаксический подход к распознаванию образов

└ Обзор подходов к задаче распознавания образов

Распознавание образов: реляционный подход

② Реляционный подход — анализ формальных понятий (АФП): формализация понятия в виде пары объём-содержание.
 $(\{1,2,3,4\}, \emptyset)$

Математический аппарат —
теория решёток



	$G \setminus M$	a	b	c	d
1		x			x
2		x		x	
3			x	x	
4			x	x	x

a — ровно 3 вершины,
b — ровно 4 вершины,
c — имеет прямой угол,
d — все стороны равны

Распознавание образов: синтаксический подход

③ Синтаксический (структурный, лингвистический) подход.

Структурный подход применяется к задачам, в которых:

- 1) важна информация об объекте, как состоящего из элементарных частей, связанных между собой определённым образом;
- 2) требуется, чтобы она давала возможность не только классифицировать объект, но и причины, которые исключают его отнесение к другому классу.

Примеры — распознавание

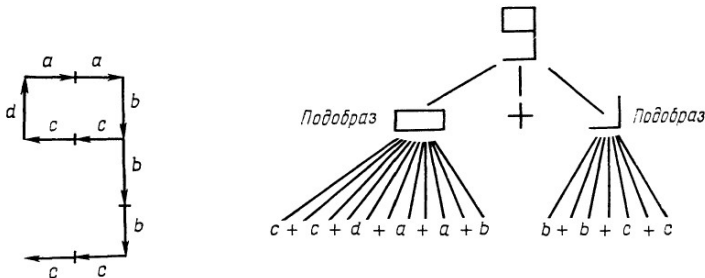
- ▶ изображений и анализ сцен: объекты обычно сложны, число требуемых признаков часто велико ⇒ удобство описания их через подобразы;
- ▶ сигналов: акустических (речь), электрических (ЭКГ, ЭЭГ, ...), электромагнитных (отражённые радиолокационные), оптических (изображения), ...

└ Введение в синтаксический подход к распознаванию образов

└ Обзор подходов к задаче распознавания образов

Распознавание образов: синтаксический подход...

Объект представляется состоящим из неделимых частей — примитивов, объединяющихся по определённым правилам.



Разные правила порождают разные классы объектов.

Лингвистическая интерпретация: аналогия между структурой объекта и синтаксисом языка даёт возможность использовать аппарат *математической лингвистики*.

└ Введение в синтаксический подход к распознаванию образов

└ Обзор подходов к задаче распознавания образов

Распознавание образов: синтаксический подход...

При дискриминантном подходе надо

- 1) сформировать признаковое пространство;
- 2) для каждого класса иметь достаточное число прецедентов.

При синтаксическом подходе надо задать правило построения классов из примитивных элементов.

Понятно, что синтаксический подход эффективен, когда:

- 1) имеется небольшое количество примитивов (элементарных подобъектов) которые легко вычленяются;
- 2) построение образов из примитивов описывается сравнительно простыми правилами.

Синтаксическое распознавание: составляющие подхода

На примере распознавания/обработки изображений.

① Предобработка:

- ▶ кодирование и аппроксимация — представление в виде, удобном для дальнейшей обработки.

Примеры:

- ▶ бинаризация черно-белых изображений,
 - ▶ дискретизация непрерывного сигнала,
 - ▶ представление функции конечным набором коэффициентов разложения по некоторой системе функций (Фурье, Уолша, ...).
-
- ▶ фильтрации, восстановление и улучшения объекта — ликвидация шума, восстановления искажений, улучшения качества.

└ Введение в синтаксический подход к распознаванию образов

└ Обзор подходов к задаче распознавания образов

Синтаксическое распознавание: составляющие подхода...

② Построение описания объекта на основе заранее заданных синтаксических операций:

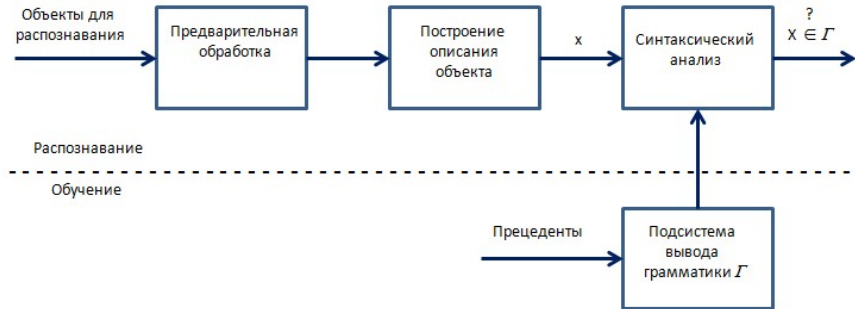
- ▶ выделение примитивов — элементарных частей объектов;
- ▶ представление объекта в виде композиции примитивов в соответствии с той или иной структурой.

③ Синтаксический анализ (грамматический разбор) — получение синтаксического описания объекта через примитивы — построение дерева грамматического разбора;

- └ Введение в синтаксический подход к распознаванию образов

- └ Обзор подходов к задаче распознавания образов

Синтаксическое распознавание: схема подхода



Структурный подход обеспечивает высокое быстродействие: распознавание сводится к сравнению символьных описаний структур, а не исходных изображений \Rightarrow преимущество при поиске в больших коллекциях графических документов.

- └ Введение в синтаксический подход к распознаванию образов

- └ Методы предобработки объектов

Разделы

Введение в синтаксический подход к распознаванию образов

Обзор подходов к задаче распознавания образов

Методы предобработки объектов

Языки описания образов

Автоматные языки и конечные автоматы

- └ Введение в синтаксический подход к распознаванию образов

- └ Методы преобработки объектов

Методы преобработки: кодирование и аппроксимация

1. Дискретизация аналоговых данных — взятие измерений (отсчетов) сигнала в дискретные моменты времени.

Теорема (Котельникова-Найквиста-Шеннона)

Если преобразование Фурье функции $f(t)$

$$\mathcal{F}\{f(t)\} = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} f(t)e^{-j\omega t} dt$$

равно нулю вне интервала $-\omega_{max} \leq \omega \leq +\omega_{max}$, то функция $f(t)$ может быть точно восстановлена по её значениям, взятым на расстоянии интервала дискретизации

$\Delta t \leq \Delta t_{max} = 1/(2f_{max})$, $\omega = 2\pi f$ друг от друга в виде

sinc-ряды Котельникова:

$$f(t) = \sum_{n=-\infty}^{+\infty} f(n \Delta t) \cdot \text{sinc} \left(\frac{t}{\Delta t} - n \right), \quad \text{sinc } x = \frac{\sin \pi x}{\pi x}.$$

Теорема Котельникова: замечания

- ▶ Теорема Котельникова применима лишь для сигналов с ограниченным спектром, т.е. принципиально для сигналов бесконечных во времени.
- ▶ На практике: полагая $\hat{\mathcal{F}} = 0$, когда $\mathcal{F}\{f(t)\} \approx 0 \Rightarrow$, при восстановлении функции по отсчётам будет получена лишь аппроксимация $f(t)$.
- ▶ Локальная интерполяция (по ограниченному числу точек) и недостаточная частота дискретизации вносит высокочастотные искажения в виде периодических копий полезной полосы частот (алиасинг, aliasing).

Теорема Котельникова: замечания...

- ▶ Сигнал невозможно восстанавливать по мере его получения и, следовательно
 - ▶ высококачественная дискредитация sinc-рядом возможна только при offline-обработке;
 - ▶ при online-обработке приходится запоминать некоторое количество отсчетов и производить интерполяцию с незначительной задержкой.
- ▶ Имеется аналогичная теорема для двумерного случая — функции $f(x, y)$.

Теорема Котельникова: рекомендации

На практике ЦОС необходимо придерживаться следующих основных рекомендаций:

- ▶ частота дискретизации должна выбираться так, чтобы она не менее чем в 2-3 раза превышала верхнюю частоту информативных составляющих сигнала (oversampling);
- ▶ предварительно сигнал следует отфильтровать фильтром нижних частот с частотой среза намного меньшей, чем половина выбранной частоты дискретизации.

└ Введение в синтаксический подход к распознаванию образов

└ Методы предобработки объектов

Методы предобработки...

2. Квантование — присваивание цифровому отсчету значения, соответствующего некоторому фиксированному уровню сигнала. *Уровень квантования* — уровень, превышение которого по абсолютной величине приводит к переходу на следующий шаг квантования: $f(t) \approx \hat{f}(t) \in \{\text{уровни квантования}\}$, которые могут располагаться

- ▶ равномерно — простота представления; при шаге квантования a значения ошибки квантования:

$$\text{максимальная} — \pm a/2; \quad \text{с.к.о.} — \frac{a}{\sqrt{12}} \approx 0,29a.$$

- ▶ неравномерно — выигрыш в точности представления: если, например, уровни яркости изображения в определенном диапазоне встречаются чаще, чем в других диапазонах, то при более плотном расположении в нём уровней квантования, точность представления увеличится.

Методы предобработки: кодирование квантованных сигналов I

После дискретизации и квантования объекта уровень квантования в каждой точке отсчета кодируют.

Характеристики кода:

- ▶ основание кода q — число цифр, букв, из которых строится код (обычно $q = 2$);
- ▶ m — число возможных уровней квантования; для представления любого заданного уровня необходимо по крайней мере $\log_2 m$ бит.

Если вероятность появления каждого из возможных уровней известна заранее, то можно построить более эффективный код, снизив среднее число бит на один уровень. Основная идея: использовать короткие коды для наиболее часто встречающихся уровней квантования.

Методы предобработки: кодирование квантованных сигналов II

- ▶ длина кода n , число информационных разрядов k , число проверочных разрядов $m = n - k$;
- ▶ избыточность кода $R = m/n$ (для делимых блочных кодов);
- ▶ кодовое расстояние d — минимальное число разрядов с различными символами;
- ▶ характеристики помехоустойчивости избыточных кодов, вероятности необнаружения ошибки и т.д.

└ Введение в синтаксический подход к распознаванию образов

└ Методы предобработки объектов

Методы предобработки: фильтрация

3. Фильтрация, восстановление и улучшение сигнала — изменение его частотного состава (спектра).

Результат — повышение качества данных \Rightarrow облегчение обнаружения заданных элементов (сигнала, изображения).

Для фильтрации используют операции, инвариантные ко времени и к изменению положения: «сглаживание», «повышение контраста», ...

Свёртка с окном $g(x, y)$, $(u, v) \notin D \Rightarrow g(x, y) = 0$:

$$f(x, y) \mapsto f_g(x, y) = (f * g)(x, y) = \int_{(u,v) \in D} g(x-u, y-v) f(u, v) du dv$$

- └ Введение в синтаксический подход к распознаванию образов

- └ Методы предобработки объектов

Методы предобработки: сглаживание...

1. Сглаживание — подавление шума, усреднение значений по некоторой окрестности

$$g(x, y) = \frac{1}{|D|}, \quad g(x, y) = \frac{1}{div} \cdot \begin{array}{|c|c|c|} \hline 1/2 & 3/4 & 1/2 \\ \hline 3/4 & 1 & 3/4 \\ \hline 1/2 & 3/4 & 1/2 \\ \hline \end{array}, \quad div = 6$$

div — коэффициент нормирования, для того чтобы средняя интенсивность оставалась не изменой.

2. Повышение контраста (дифференцирование)

$\nabla f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$ — свёртка с окном

$$g(x, y) = \begin{array}{|c|c|c|} \hline -1 & -2 & -1 \\ \hline -2 & +12 & -2 \\ \hline -1 & -2 & -1 \\ \hline \end{array}, \quad g(x, y) = \begin{array}{|c|c|c|} \hline 0 & -1 & 0 \\ \hline -1 & +5 & -1 \\ \hline 0 & -1 & 0 \\ \hline \end{array}$$

└ Введение в синтаксический подход к распознаванию образов

└ Методы предобработки объектов

Методы предобработки: фильтрация...

Для дискретных сигналов применяют *линейную цифровую фильтрацию*, задаваемую *обобщенным разностным уравнением цифрового фильтра*:

$$y(n) = \sum_{i \in I} b_i x(n - i) + \sum_{j \in J} a_j y(n - j),$$

где $x(n)$ и $y(n)$ — соответственно входной и выходной сигнал,
 b_i, a_j — коэффициенты фильтра,
 I, J — интервалы \mathbb{Z} , содержащие 0, *параметры фильтра*.

Частные случаи:

- $a_j = 0$ для всех $j \in J$ — *нерекурсивный* фильтр, иначе — *рекурсивный*;
- все коэффициенты фильтра суть константы — *фильтр с постоянными коэффициентами*, иначе — *адаптивный*.

Цифровые фильтры: примеры

1) Сглаживание пятёрками —

$$y(n) = \frac{1}{5} [x(n-2) + x(n-1) + x(n) + x(n+1) + x(n+2)]$$

2) Фильтр нижних частот (*фильтр Хемминга*) —

$$y(n) = \frac{1}{8}x(n) + \frac{1}{4}x(n-1) + \frac{1}{4}x(n-2) + \frac{1}{4}x(n-3) + \frac{1}{8}x(n-4)$$

3) Сглаживающий дифференциатор —

$$y(n) = \frac{1}{8} [x(n-8) + x(n-7) + x(n-6) + x(n-5) - \\ - x(n-3) - x(n-2) - x(n-1) - x(n)]$$

4) Численное интегрирование по методу трапеций —

$$y(n) = y(n-1) + \frac{1}{2}x(n) + \frac{1}{2}x(n-1)$$

└ Введение в синтаксический подход к распознаванию образов

└ Методы предобработки объектов

Предобработка: сегментация изображений

4. Сегментация — разбиение изображения на фрагменты, не похожие по какому-то признаку.

Сегментация разделяет изображение на составляющие части или объекты, а уровень детализации разделяемых областей зависит от решаемой задачи \Rightarrow универсального метода сегментации не существует.

Задачи автоматической сегментации изображений:

- 1) выделение областей с известными свойствами;
- 2) разбиение изображения на однородные области.

Предобработка: методы сегментации изображений

— базируются на распределении яркостей точек изображения.

① Пороговые методы —

используют *пороговую операцию*: фрагмент есть совокупность точек, яркость которых больше/меньше порога/лежит между пороговыми значениями).

Например, в задаче распознавания знаков на бумаге, сами знаки представлены совокупностью черных, а бумага (фон) — совокупностью белых точек \Rightarrow отделить знак от фона можно при помощи пороговой операции.

Существенная трудность — выбор значения порога.

Пороговые методы сегментации изображений

► *Метод процентилей.*

Если определена приблизительная площадь S фрагмента, то выбирают пороги так, чтобы площадь выделяемого сегмента с яркостью меньше/больше порога была приблизительно равна S .

► *Метод моды.*

Если определен интервал яркости I фрагмента, то выбирают пороги так, чтобы в сегмент попали точки с яркостью из I .

Например, если яркости фрагмента имеет пик, то границами I могут быть локальные минимумы распределения, расположенным по обе стороны от этого пика.

Пороговые методы сегментации изображений...

- ▶ *Метод Оцу (Otsu)* вычисления порога бинаризации полутонового изображения: ищется порог h , минимизирующий дисперсию внутри классов:

$$h = \arg \min_t \{ \omega_1(t) \sigma_1^2(t) + \omega_2(t) \sigma_2^2(t) \},$$

где веса ω_i — вероятности (доли) двух классов, разделенных порогом t , σ_i^2 — дисперсии классов, $i = 1, 2$.

Минимизация дисперсии внутри класса = максимизация дисперсии между классами.

└ Введение в синтаксический подход к распознаванию образов

└ Методы предобработки объектов

Пороговые методы сегментации изображений...

- ▶ *Прослеживание границ (контуров, краёв, ...).*

Например,

- ▶ простой способ отслеживания границ на двухградационном изображении: просмотр изображения, пока не встретится пара точек раstra разной яркости,
- ▶ управляемые (steerable) фильтры, осуществляющие дифференцирование по направлению.

Для повышения устойчивости к шуму, перед применением фильтрации изображение обычно размывают.

Пороговые методы сегментации изображений...

Пример оператора пространственного дифференцирования для формирования контурных изображений — *оператор Собела*, имеющий наименьший коэффициент утолщения контурной линии.

$f(x, y) \rightarrow g(x, y) = \|\nabla f(x, y)\| = \sqrt{d_1^2 + d_2^2}$ — модуль градиента

d_1, d_2 — суммы элементов свёртки $f(x, y)$ с окнами H_1 и H_2 соответственно

$$H_1 = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}, \quad H_2 = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}.$$

«Не-пороговые» методы сегментации изображений

② «Не-пороговые» методы

▶ *Выращивание регионов, дробление-слияние*

— последовательная проверка точек на их принадлежность к фрагменту.

Методы этой группы учитывают пространственное расположение точек напрямую и позволяют:

- ▶ выделять фрагменты;
- ▶ проследивать границы или контуры фрагментов;
- ▶ получать характеристики фрагментов (площадь, параметры формы, ...).

Методы позволяют варьировать значение порога в зависимости от характера уже проверенных точки. Результат такой процедуры зависит от того, какая последовательность будет выбрана.

«Не-пороговые» методы сегментации изображений...

- ▶ *Моделирование изображения случайным полем*
— в основе предположение, что характеристики каждой точки изображения зависит от характеристик некоторого множества соседних точек.
Как правило, рассматривают марковское случайное поле.
- ▶ *Методы, основанные на теории графов*
Общая идея:
 - ▶ изображение представляется в виде взвешенного графа, с вершинами в точках изображения,
 - ▶ вес ребра графа отражает сходство точек в некотором смысле (расстояние между точками по некоторой метрике),
 - ▶ разбиение изображения моделируется разрезами графа.

└ Введение в синтаксический подход к распознаванию образов

└ Методы предобработки объектов

«Не-пороговые» методы сегментации изображений...

▶ *Оптимизационный подход*

— задачу разбиения изображения на однородные области сводят к задаче оптимизации некоторого функционала от разбиения.

Например, в графовых методах оптимизируется функционал стоимости разреза.

▶ *Настройка параметров*

— одно из наиболее перспективных на данный момент направлений.

Почти у всех вышеописанных методов сегментации есть масса параметров, значения которых для каждой конкретной задачи приходится подбирать эвристически. Одним из подходов к решению этой проблемы является использование машинного обучения.

- └ Введение в синтаксический подход к распознаванию образов

- └ Языки описания образов

Разделы

Введение в синтаксический подход к распознаванию образов

- Обзор подходов к задаче распознавания образов

- Методы предобработки объектов

- Языки описания образов

Автоматные языки и конечные автоматы

Выбор терминальных элементов

— первый этап построения синтаксической модели образов.

Требования к терминальным элементам (примитивам):

- ▶ они должны служить основными элементами образов и обеспечивать адекватное описание объекта в терминах заданных структурных отношений;
- ▶ их выделение и распознавание должны легко осуществляться.

Эти требования часто противоречат друг другу.

Примеры производных элементов:

- ▶ для речевых образов — совокупность фонем (= звуков, минимальных различимых единиц языка).
- ▶ для рукописного текста — штрихи.

Разные постановки задачи обуславливают выбор разных производных элементов.

Языки и порождающие грамматики (напоминание)

Определение

Порождающая грамматика G есть четверка

$$G = \langle V, W, I, R \rangle,$$

где (все множества конечны)

V — множество терминальных символов;

W — множество нетерминальных (вспомогательных) символов, $V \cap W = \emptyset$;

I — начальный символ, $I \in W$;

R — совокупность правил вывода (подстановок цепочек символов).

Язык $L(G)$, порождаемый грамматикой G —

$$L(G) = \{ x \in V^* \mid \exists \text{ вывод } I \Rightarrow x \}$$

Порождающие грамматики по форме правил подстановки разделены на 4 типа (Н.Хомский, 1957).

Грамматики типа 0 — неограниченные

Самый широкий класс: не имеет каких-либо ограничений на вид правил подстановки.

Пример ($V = \{a, b, c\}$, $W = \{I, A, B\}$)

$$R : I \rightarrow aAbc$$

$$Ab \rightarrow bA$$

$$Ac \rightarrow Bbcc$$

$$bB \rightarrow Bb$$

$$aB \rightarrow aaA$$

$$aB \rightarrow \quad \text{— пустой символ}$$

Грамматика порождает слова вида $x = a^n b^{n+2} c^{n+2}$, $n \geq 0$.

Пример: $I \Rightarrow aAbc \Rightarrow abAc \Rightarrow abBbcc \Rightarrow aBbbcc \Rightarrow bbcc$.

Для использования этот класс слишком широк.

└ Введение в синтаксический подход к распознаванию образов

└ Языки описания образов

Грамматики типа 1 — непосредственно составляющих или *контекстно-зависимые*.

Имеют правила вида $\alpha_1 A \alpha_2 \rightarrow \alpha_1 \beta \alpha_2$,

где $A \in W$, $\beta \in T^+$, $\alpha_1, \alpha_2 \in T^*$, $T = V \cup W$.

Пример ($V = \{a, b, c\}$, $W = \{I, A, B\}$)

$$R : I \rightarrow abc$$

$$I \rightarrow aAbc$$

$$Ab \rightarrow bA$$

$$Ac \rightarrow aaA$$

$$aB \rightarrow aa$$

Эта грамматика непосредственно составляющих порождает слова вида $x = a^n b^n c^n$, $n \geq 1$.

└ Введение в синтаксический подход к распознаванию образов

└ Языки описания образов

Грамматики типа 2 — *бесконтекстные* или *контекстно-свободные, КС*.

Имеют правила подстановки вида $A \rightarrow \beta$, где $A \in W$, $\beta \in T^+$, т.е. заменяют вспомогательный символ A на непустую цепочку произвольных символов вне зависимости от контекста (окружения A).

Пример ($V = \{a, b\}$, $W = \{I\}$)

$$R : I \rightarrow ab$$

$$I \rightarrow aIb$$

Эта бесконтекстная грамматика порождает слова вида $x = a^n b^n$, $n \geq 1$.

Граматики типа 2...

Другим методом описания вывода в бесконтекстной грамматике является использование *деревьев вывода*.

Построение дерева вывода:

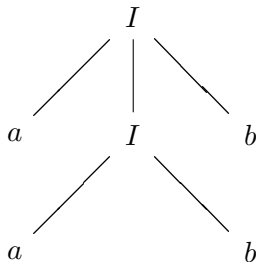
- ▶ каждая вершина дерева имеет метку — символ из алфавита T , корень дерева имеет метку I .
- ▶ если вершина с меткой A — не лист дерева, то $A \in W$.
- ▶ если вершине n с меткой A прямо подчинены вершины n_1, \dots, n_k , помеченные (слева направо) A_1, \dots, A_k , то в R должно быть правило $A \rightarrow A_1, \dots, A_k$.

- └ Введение в синтаксический подход к распознаванию образов

- └ Языки описания образов

Грамматики типа 2...

Вывод слова $aabb$:



└ Введение в синтаксический подход к распознаванию образов

└ Языки описания образов

Граматики типа 3 — *автоматные или регулярные*

Имеют правила подстановок вида $A \rightarrow aB$ или $A \rightarrow a$,
где $A, B \in W$.

Это самые простые из формальных грамматик; они являются контекстно-свободными, но с ограниченными возможностями.

Пример ($V = \{a, b\}$, $W = \{I, A\}$)

$$R : I \rightarrow aA$$

$$A \rightarrow aA$$

$$A \rightarrow b$$

Эта регулярная грамматика порождает слова вида $x = a^n b$, $n \geq 1$.

$$G_3 \subset G_2 \subset G_1 \subset G_0$$

Программные грамматики

— ограничивает выбор следующего правила подстановки.

Определение

Программная грамматика G_p есть пятёрка

$$G_p = \langle V, W, I, P, J \rangle,$$

где (все множества конечны)

V — множество терминальных символов;

W — множество нетерминальных символов;

I — начальный символ, $I \in W$;

P — множество правил вывода (подстановки);

J — множество меток правил подстановки, обычно

$J = \{1, \dots, m\}$ — номер правила.

Программные грамматики: определение

Правило вывода из P имеет вид

$$(r) \quad \alpha \rightarrow \beta \quad S \quad F,$$

где $r \in J$ — метка,

*$S, F \subseteq J$ — два списка меток переходов при успехе
и неудаче соответственно,*

*$\alpha \rightarrow \beta$ — правило подстановки (ядро),
 $\alpha \in T^*WT^*$, $\beta \in T^*$.*

Программная грамматика действует следующим образом:

- ▶ сначала применяется правило с меткой (1);
- ▶ если делается попытка применить правило (r) , то при невозможности его применения следующее выбирается из списка F , а в противном случае его выполнения следующее выбирается из списка S .

└ Введение в синтаксический подход к распознаванию образов

└ Языки описания образов

Примеры двумерных грамматик. 1. Домики

① Грамматика, описывающая изображения домиков:

$$G = \langle V, W, I, R \rangle.$$

$$V = \{ \langle \text{домик} \rangle, \langle \text{профиль} \rangle, \langle \text{фасад} \rangle, \langle \text{крыша} \rangle, \langle \text{фронтон} \rangle, \\ \langle \text{стена} \rangle, \langle \text{дымоход} \rangle, \langle \text{окна} \rangle, \langle \text{дверь} \rangle \}, \quad I = \langle \text{домик} \rangle$$

$$W = \{ \square, \sqcap, \blacksquare, \boxplus, \triangle, \square, \nabla, \rightarrow, (, \odot, \ominus, \uparrow, \mapsto \}$$

Обозначения:

\rightarrow (XY) X — справа от Y,

\odot (XY) X — внутри Y,

\ominus (XY) X — внутри Y и внизу,

\uparrow (XY) X — лежит сверху Y,

\mapsto (XY) X — прилегает к Y справа.

└ Введение в синтаксический подход к распознаванию образов

└ Языки описания образов

Примеры двумерных грамматик. 1. Домики $R =$

$\langle \text{дверь} \rangle \rightarrow \text{▣}$

$\langle \text{окна} \rangle \rightarrow \text{▣} , \langle \text{окна} \rangle \rightarrow \rightarrow (\langle \text{окна} \rangle, \text{▣})$

$\langle \text{дымоход} \rangle \rightarrow \text{▣} , \langle \text{дымоход} \rangle \rightarrow \text{▣}$

$\langle \text{стена} \rangle \rightarrow \text{ } , \langle \text{стена} \rangle \rightarrow \text{○} (\langle \text{дверь} \rangle, \text{▣})$

$\langle \text{стена} \rangle \rightarrow \text{⊙} (\langle \text{окна} \rangle, \text{▣})$

$\langle \text{фронтон} \rangle \rightarrow \text{△} , \langle \text{фронтон} \rangle \rightarrow \uparrow (\langle \text{дымоход} \rangle, \text{△})$

$\langle \text{крыша} \rangle \rightarrow \text{▤} , \langle \text{крыша} \rangle \rightarrow \uparrow (\langle \text{дымоход} \rangle, \text{▤})$

$\langle \text{фасад} \rangle \rightarrow \uparrow (\langle \text{фронтон} \rangle, \langle \text{стена} \rangle)$

$\langle \text{профиль} \rangle \rightarrow \uparrow (\langle \text{крыша} \rangle, \langle \text{стена} \rangle)$

$\langle \text{домик} \rangle \rightarrow \langle \text{фасад} \rangle,$




$\langle \text{домик} \rangle \rightarrow \rightarrow (\langle \text{домик} \rangle, \langle \text{профиль} \rangle).$

- └ Введение в синтаксический подход к распознаванию образов

- └ Языки описания образов

Примеры двумерных грамматик...

Описание трёх изображений:

<i>Домик</i>	<i>Описание</i>
	$\uparrow (\triangle, \square)$
	$\uparrow (\uparrow (\nabla, \triangle), \odot (\square, \square))$
	$\rightarrow (\uparrow (\uparrow (\square, \nabla), \odot (\square, \square)), \uparrow (\triangle, \odot (\square, \square)))$

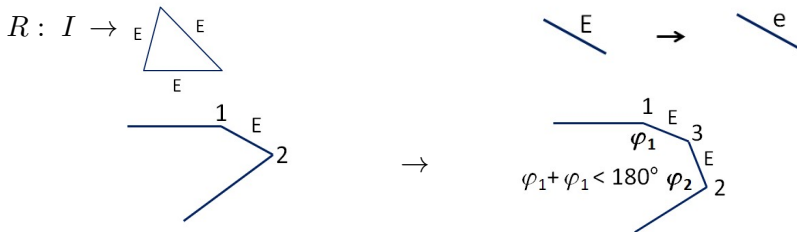
└ Введение в синтаксический подход к распознаванию образов

└ Языки описания образов

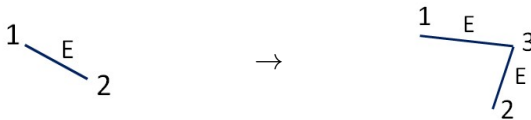
Примеры двумерных грамматик. 2. Многоугольники

② Грамматика, описывающая многоугольники:

1) выпуклые —



2) общего вида — замена последнего правила



└ Введение в синтаксический подход к распознаванию образов

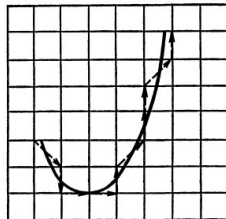
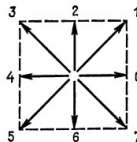
└ Языки описания образов

Выделение терминальных элементов

Методы выделения терминальных элементов на изображении:
упор на 1) границы; 2) области.

1. Выделение терминальных элементов на основе границ —
схема *цепного кодирования*:

- ▶ на изображение накладывают сетку и узлы сетки, наиболее близкие к точкам изображения соединяют отрезками прямых;
- ▶ каждому полученному отрезку присваивают восьмеричное число



— изображение представляется последовательностью/ями
восьмеричных чисел (7600212212).

└ Введение в синтаксический подход к распознаванию образов

└ Языки описания образов

Выделение терминальных элементов: цепное кодирование

Свойства цепного кодирования:

- ▶ поворот на угол, кратный 45° , сводится к $+8$;
- ▶ легко выполнимы растяжение, определение длины кривой, самопересечений...
- ▶ применимо для границ произвольной связности.

Метод применялся для

- ▶ распознавания рукопечатного текста;
- ▶ классификации фотографий треков частиц в пузырьковой и искровой камерах;
- ▶ анализе хромосом;
- ▶ идентификации отпечатков пальцев
- ▶ ...

└ Введение в синтаксический подход к распознаванию образов

└ Языки описания образов

Выделение терминальных элементов

2. Выделение терминальных элементов на основе областей

Терминальные элементы — полуплоскости: выпуклые многоугольники представляются пересечением конечного числа полуплоскостей.

Полуплоскости \rightarrow буквы,

Слова \rightarrow выпуклые многоугольники,

Предложения \rightarrow фигуры.

- H_1, \dots, H_k — полуплоскости с заданными заранее направлениями $\varphi_1, \dots, \varphi_k$;
- G — группа параллельных переносов;
- первичное множество

$$R_j = \bigcap_{i=1}^{2k} g_i^j H_i, \quad g_i^j \in G;$$

- многоугольник

$$A = \bigcup_{j=1}^l R_j.$$

└ Введение в синтаксический подход к распознаванию образов

└ Языки описания образов

Построение грамматики

Пример. Построить грамматики для регулярного языка $L = \{a^n, b^n, c^n \mid 1 \leq n \leq 3\}$, описывающий равносторонние треугольники с длиной стороны 1, 2, 3.

1. Автоматная грамматика $G_3 = \langle V, W, I, R \rangle$, где $W = \{I, A_1, A_2, B_{10}, B_{20}, B_{30}, B_{21}, B_{31}, B_{32}, C_1, C_2, C_3\}$,

$V = \{ \xrightarrow{a}, \begin{array}{c} \swarrow b \\ \searrow c \end{array}, \}$, R :

$$I \rightarrow aA_1 \quad B_{10} \rightarrow bC_1 \quad B_{32} \rightarrow bC_3$$

$$I \rightarrow aB_{10} \quad B_{20} \rightarrow bB_{21} \quad C_1 \rightarrow c$$

$$A_1 \rightarrow aA_2 \quad B_{21} \rightarrow bC_2 \quad C_2 \rightarrow cC_1$$

$$A_1 \rightarrow aB_{20} \quad B_{30} \rightarrow bB_{31} \quad C_3 \rightarrow cC_2$$

$$A_2 \rightarrow aB_{30} \quad B_{31} \rightarrow bB_{32}$$

└ Введение в синтаксический подход к распознаванию образов

└ Языки описания образов

Построение грамматики...

2. Бесконтекстная грамматика $G_2 = \langle V, W, I, R \rangle$, где

$W = \{ I, A_1, A_2, B_1, B_2, B_3, C \}$, $V = \{ a, b, c \}$,

R :

$I \rightarrow aA_1C$ $A_1 \rightarrow aA_2C$ $B_2 \rightarrow bB_1$

$A_1 \rightarrow b$ $A_2 \rightarrow aB_3C$ $B_1 \rightarrow b$

$A_1 \rightarrow aB_2C$ $B_3 \rightarrow bB_2$ $C \rightarrow c$

Беконтекстная грамматика G_2 гораздо компактнее автоматной G_3 .

Грамматика непосредственно оставляющих для данного языка (G_1 , не приведена) мало отличается от беконтекстной G_2 .

- └ Введение в синтаксический подход к распознаванию образов

- └ Языки описания образов

Построение грамматики...

3) Программная грамматика $G_p = \langle V, W, I, P, J \rangle$, где $W = \{I, B, C\}$, $V = \{a, b, c\}$, $J = \{1, \dots, 5\}$, P :

Метка	Ядро	I	F
1	$I \rightarrow aB$	$\{2, 3\}$	$\{\emptyset\}$
2	$B \rightarrow aBB$	$\{2, 3\}$	$\{\emptyset\}$
3	$B \rightarrow C$	$\{4\}$	$\{5\}$
4	$C \rightarrow bC$	$\{3\}$	$\{\emptyset\}$
5	$C \rightarrow c$	$\{5\}$	$\{\emptyset\}$

Все правила имеют вид $A \rightarrow \beta$, $A \in W$, $\beta \in T^*$, такая программная грамматика называется *бесконтекстной*.

Бесконтекстная программная грамматика G_p описывает язык $L = \{a^n, b^n, c^n \mid n = 1, 2, \dots\}$ и ещё более компактна, чем, бесконтекстная G_2 .

Разделы

Введение в синтаксический подход к распознаванию образов

Обзор подходов к задаче распознавания образов

Методы предобработки объектов

Языки описания образов

Автоматные языки и конечные автоматы

Альтернативный способ задания автоматной грамматики

— конечным недетерминированным автоматом, генерирующим регулярный язык.

Определение

Конечный недетерминированный автомат \tilde{A} есть пятёрка $\tilde{A} = \langle X, K, \varphi, P, \psi \rangle$, где (все множества конечны):

X — выходной алфавит (символы X генерируются автоматом \tilde{A}),

K — множество состояний автомата,

φ — предикат на K , определяющий начальные состояния,

ψ — предикат на K , определяющий финальные состояния,

P — предикат на $K \times X \times K$, определяющий многозначную функцию переходов: если в $(i - 1)$ -й момент времени автомат находился в состоянии k' и $P(k', x, k) = 1$, то в i -й момент автомат может выдать символ x и перейти в состояние k .

Язык $L(\tilde{A})$ автомата \tilde{A}

— множество слов, которые могут появиться на его выходе:

$\bar{x} = (x_1, x_2, \dots, x_n) \in L(\tilde{A})$, если существует такая последовательность (k_0, k_1, \dots, k_n) состояний \tilde{A} , что

- 1) $\varphi(k_0) = 1$ (k_0 — начальное состояние);
- 2) $P(k_{i-1}, x_i, k_i) = 1$, $i = \overline{1, n}$ (переход $k_{i-1} \rightarrow k_i$ возможен для любого i);
- 3) $\psi(k_n) = 1$ (k_n — финальное состояние).

Это эквивалентно истинности предиката

$$F(\bar{x}) = \bigvee_{k_0 \in K} \dots \bigvee_{k_n \in K} \varphi(k_0) \& \bigg\& \bigg\limits_{i=1}^n P(k_{i-1}, x_i, k_i) \& \psi(k_n),$$

где $F(\bar{x})$ — утверждение «слово $\bar{x} = (x_1, \dots, x_n)$ принадлежит языку $L(\tilde{A})$ ».

Компактная запись предиката $F(\bar{x})$

— в виде матричного произведения:

$\varphi = (\varphi_1, \dots, \varphi_{|K|})$ — вектор-строка, $\varphi_k = \varphi(k)$,

$\psi = (\psi_1, \dots, \psi_{|K|})$ — вектор-столбец, $\psi_k = \psi(k)$,

$P_i = \|P(k', x_i, k)\|$ — квадратная порядка $|K|$ матрица переходов, отвечающая выходному символу x_i , $i = \overline{1, n}$ (ясно, что существует всего $|X|$ различных матриц P_i),

⊗ — матричное произведение с операциями сложения \vee и умножения $\&$, тогда

$$F(\bar{x}) = \varphi \otimes \bigotimes_{i=1}^n P_i \otimes \psi.$$

Сложность вычисления этого предиката — $O(|K|^{2n})$.

Цель распознавания — определить

$\bar{x} = (x_1, x_2, \dots, x_n) \stackrel{?}{\in} L(\mathcal{A})$; для этого нужен *распознающий автомат*.

Задачу « $\bar{x} \in L(\tilde{\mathcal{A}})$?» решает распознающий автомат

Определение

Конечный детерминированный автомат \mathcal{A} есть пятёрка $\langle X, K, k_0, q, K' \rangle$, где (все множества конечны)

X — входной алфавит (символы X подаются на вход автомата \mathcal{A}),

K — множество состояний автомата,

$k_0 \in K$ — одно из начальных состояний,

$q : K \times X \rightarrow K$ — функция переходов,

$K' \subset K$ — множество финальных состояний.

Если в $(i - 1)$ -й момент времени \mathcal{A} находился в состоянии k , а в i -й момент на его вход подан символ x , то \mathcal{A} окажется в состоянии $q(x, k) \Rightarrow$ слово x_1, x_2, \dots, x_n , поданное на вход \mathcal{A} однозначно определит последовательность его состояний $k_0, k_1 = q(k_0, x_1), \dots, k_n = q(k_{i-1}, x_n)$.

Автомат \mathcal{A} распознаёт принадлежность слова \bar{x} языку $L(\mathcal{A})$, если $k_n \in K'$.

Генерирующий автомат как грамматика

Рассмотрим генерирующий автомат $\tilde{A} = \langle X, K, \varphi, P, \psi \rangle$.

- ▶ Назовём X и K — терминальным и нетерминальным алфавитами соответственно.
- ▶ Функцию φ представим в виде подмножества начальных состояний (аксиом) $K^0 = \{k \in K \mid \varphi(k) = 1\}$.
- ▶ Функции P и ψ выразим в виде подмножеств троек (k', x, k'') и пар (k, x) , $k, k', k'' \in K$, $x \in X$ соответственно.
- ▶ Если справедливо $P(k', x, k'') = 1$, то тройку (k', x, k'')
 - ▶ запишем в виде правила $k' \rightarrow xk''$,
 - ▶ а если при этом справедливо ещё и $\psi(k'') = 1$, то вводим ещё и правило $k' \rightarrow x$.

Множество правил обозначим R .

Генерирующий автомат как грамматика...

В результате автомат $\tilde{\mathcal{A}}$ представляется в виде регулярной (автоматной) грамматики как четвёрка

$$G(\mathcal{A}) = \langle X, K, K^0, R \rangle.$$

Регулярная грамматика определяет язык, т.е.сть $L \subset X^*$, следующим образом:

- 1) слово, состоящая из единственного символа, который является одной из аксиом, называется выведенным в данной грамматике;
- 2) если слово $\bar{x}k'$, $\bar{x} \in X^*$, $k' \in K$ выведено, а множество правил содержит правило $k' \rightarrow x'k''$, то слово $\bar{x}x'k''$ тоже считается выведенным;
- 2) если слово $\bar{x}k'$, $\bar{x} \in X^*$, $k' \in K$ выведено, а множество правил содержит правило $k' \rightarrow x'$, то слово $\bar{x}x'$ принадлежит языку L , порождаемому данной грамматикой.

Генерирующий автомат как грамматика...

Автомат \tilde{A} и грамматика $G(\tilde{A})$ порождают один и тот же язык $L \subset X^*$ (X^* — рефлексивное замыкание алфавита $X =$ множество всех предложений конечной длины, включая пустое, в алфавите X).

Возможен и обратный переход — от регулярной грамматики к автомату.

Недетерминированный конечный автомат и регулярная грамматика — два эквивалентных способа задания регулярных языков (типа 3).

Операции на множестве языков

Введем следующие три операции на множестве языков.

1. *Итерация языка L* есть язык, обозначаемый L^* и определяемый следующим образом.
 - ▶ пустое слово (слово нулевой длины), принадлежит L^* .
 - ▶ если слово \bar{x} принадлежит L^* , слово \bar{y} принадлежит L , то слово $\bar{x}\bar{y}$ также принадлежит L^* .
2. *Конкатенация языков L_1 и L_2* есть язык, обозначаемый L_1L_2 и содержащий все слова вида $\bar{x}\bar{y}$, $\bar{x} \in L_1$, $\bar{y} \in L_2$ и только их.
3. *Объединение языков L_1 и L_2* есть язык $L_1 \cup L_2$.

Регулярные выражения в алфавите X I

Регулярные языки можно также задать с помощью *регулярных выражений* по следующим правилам.

1. \emptyset — регулярное выражение, обозначающее пустое множество слов.
2. $\#$ — регулярное выражения, обозначающее язык, состоящий из единственного пустого слова (нулевой длины).
3. Для каждого символа $x \in X$ запись x — регулярное выражение, обозначающее язык, состоящий из единственного однобуквенного слова x .
4. Если α — регулярное выражение языка L , то $(\alpha)^*$ — регулярное выражение для итерации языка L^* .

Регулярные выражения в алфавите X II

5. Если α_1 и α_2 — регулярные выражение языков L_1 и L_2 соответственно, то $\alpha_1\alpha_2$ — регулярное выражение для конкатенации L_1L_2 , а α_1, α_2 — регулярное выражение для объединения $L_1 \cup L_2$.

Например, $a(b, c)^*$ — регулярное выражение, задающее множество слов

- 1) начинающихся символом a ,
- 2) вслед за которым идёт любая (возможно пустая) последовательность, составленная из символов b и c .

Пример записи регулярного языка

Задача: задать язык L , множество слов вида

$\langle \text{идентификатор} \rangle = \langle \text{« сумма произведений идентификаторов»} \rangle$,

в алфавите $X = \{a, b, c, +, \times, =\}$, где *идентификатор* — слово длины > 0 , состоящее букв a, b, c — простейшие операторы присваивания.

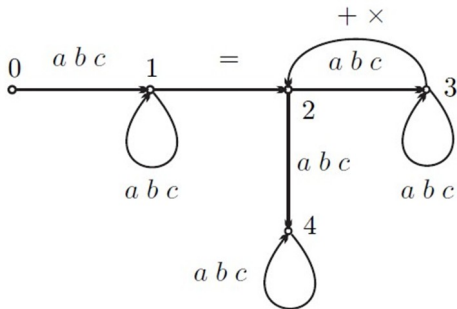
Например $x = ab + c \times a + aba \in L$, а $bc =, a = a + \times b \notin L$.

① Регулярное выражение, задающее язык L :

$$(a, b, c)(a, b, c)^* = ((a, b, c)(a, b, c)^*(+, \times))^* (a, b, c)(a, b, c)^*.$$

Пример записи регулярного языка...

② Недетерминированный автомат, порождающий язык L :



Выходной алфавит $X = \{a, b, c, +, \times, =\}$.

Множество состояний $K = \{0, 1, 2, 3, 4\}$ — множество вершин графа; начальное состояние — 0, конечное — 4.

Пример записи регулярного языка...

Если стрелке $k' \rightarrow k''$ приписан символ x , то $P(k', x, k'') = 1$ (для всех других троек функция $P = 0$), т.е. P принимает значение 1 на следующих тройках:

$(0, a, 1), (0, b, 1), (0, c, 1),$	$(3, a, 3), (3, b, 3), (3, c, 3),$
$(1, a, 1), (1, b, 1), (1, c, 1),$	$(3, +, 2), (3, \times, 2),$
$(1, =, 2),$	$(2, a, 4), (2, b, 4), (2, c, 4),$
$(2, a, 3), (2, b, 3), (2, c, 3),$	$(4, a, 4), (4, b, 4), (4, c, 4).$

Пример записи регулярного языка...

③ Язык L как регулярная грамматика:

терминальный алфавит $V = \{a, b, c, =, +, \times\}$,

нетерминальный алфавит $W = \{0, 1, 2, 3, 4\}$,

начальное состояние $I = 0$, правила R :

$0 \rightarrow a1, 0 \rightarrow b1, 0 \rightarrow c1, \quad 2 \rightarrow a4, 2 \rightarrow b4, 2 \rightarrow c4,$

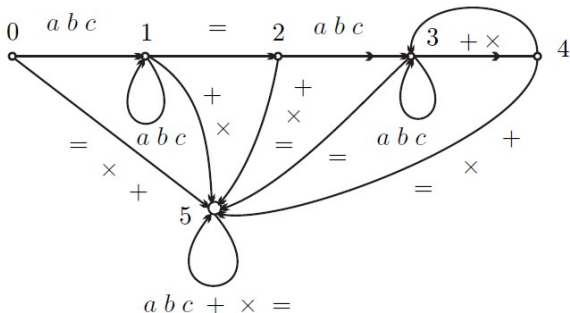
$1 \rightarrow = 2, \quad 4 \rightarrow a4, 4 \rightarrow b4, 4 \rightarrow c4,$

$2 \rightarrow a3, 2 \rightarrow b3, 2 \rightarrow c3, \quad 4 \rightarrow a, 4 \rightarrow b, 4 \rightarrow c,$

$3 \rightarrow a3, 3 \rightarrow b3, 3 \rightarrow c3.$

Пример записи регулярного языка...

④ Детерминированный автомат, распознающий язык L :



Начальное состояние $k_0 = 0$, финальное $K' = \{3\}$ (5 — поглощающее состояние).

Стрелки задают функцию переходов: если стрелка начинается в вершине k' , заканчивается в вершине k'' и на ней записан символ x , это обозначает, что $q(k', x) = k''$.

Пример записи регулярного языка...

Входной алфавит $X = \{a, b, c, =, +, \times\}$,
множество состояний $K = \{0, \dots, 5\}$.

Любое правильное слово переводит автомат в состояние 3, а любое неправильное — в какое-то другое, что достигается выбором функции переходов q .

В рамках структурного подхода задача распознавания состоит в построении алгоритма, который для любого слова \bar{x} данного регулярного языка определяет, принадлежит ли оно языку L .

Эта задача оказывается не очень трудной, если регулярный язык выражен с помощью генерирующего автомата: распознающий автомат определяется через множество последовательностей, для которых матричное произведение равно 1.

Штрафные автоматы и языки

Для формализации сложных понятий (классов) необходимо задание функции, выражающей степень уверенности принадлежности объекта классу \Rightarrow понятие штрафных автоматов и языков.

Пусть X и K — два конечных множества, а три функции

$$\varphi : K \rightarrow \mathbb{R}_{\geq 0}, \quad P : K \times X \times K \rightarrow \mathbb{R}_{\geq 0}, \quad \psi : K \rightarrow \mathbb{R}_{\geq 0}$$

определяют поведение *штрафного автомата* \mathcal{F} :

- ▶ автомат может начать свою работу в любом состоянии $k \in K$, заплатив при этом штраф $\varphi(k)$;
- ▶ если автомат в $(i - 1)$ -й момент находился в состоянии k' , то в i -й момент он может сгенерировать символ x и перейти в состояние k , заплатив штраф $P(k', x, k)$;
- ▶ автомат может завершить работу в любой момент i , заплатив за это штраф $\psi(k_i)$.

Штрафные автоматы и языки...

Штрафной язык определяется автоматом

$\mathcal{F} = \langle X, K, \varphi, P, \psi \rangle$ и числом ε : в него входят те и только слова, штраф за генерирование которых не превосходит ε .

Регулярные языки — собственное подмножество штрафных.

Множество языков, которые могут быть определены указанным способом, включают все регулярные языки, а также некоторые другие языки, не являющиеся регулярными. Распознавание принадлежности последовательности $\bar{x} = (x_1, x_2, \dots, x_n)$ заданному штрафному языку сводится к вычислению предиката

$$\min_{k_0} \min_{k_1} \dots \min_{k_n} \left\{ \varphi(k_0) + \sum_{i=1}^n P(k_{i-1}, x_i, k_i) + \psi(k_n) \right\} \stackrel{?}{\leq} \varepsilon.$$

Распознавание принадлежности последовательности штрафному языку

Полукольцо — алгебраическая структура, похожая на кольцо, но без требования существования противоположного по сложению элемента.

Пример полукольца — $(\min, +)$; здесь сложение — \min , умножение — $+$.

Обозначим в этом полукольце операцию матричного произведения \odot .

Тогда та же формула запишется как:

$$\varphi \odot \left(\bigodot_{i=1}^n P_i \right) \odot \psi \stackrel{?}{\leq} \varepsilon.$$

Задача остается простой: её формулировка указывает алгоритм решения с вычислительной сложностью $O(|K|^2 n)$.